NASA Technical Memorandum 107592

Automatic Learning Rate Adjustment for Self-Supervising Autonomous Robot Control

Michael K. Arras and Peter W. Protzel
Institute for Computer Applications
in Science and Engineering

Daniel L. Palumbo
NASA Langley Research Center

March 1992

National Aeronautics and
Space Administration

Langley Research Center
Hampton, VA 23665

Automatic Learning Rate Adjustment for Self-Supervising Autonomous Robot Control*

Michael K. Arras
Institute for Computer Applications in Science and Engineering
Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23665

Peter W. Protzel
Institute for Computer Applications in Science and Engineering
Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23665

Daniel L. Palumbo
System Validation Methods Branch
Mail Stop 130, NASA Langley Research Center, Hampton, VA 23665

Abstract

Described is an application in which an Artificial Neural Network (ANN) controls the positioning of a robot arm with five degrees of freedom by using visual feedback provided by two cameras. This application and the specific ANN model, local linear maps, are based on the work of Ritter, Martinetz, and Schulten. We extended their approach by generating a filtered, average positioning error from the continuous camera feedback and by coupling the learning rate to this error. When the network learns to position the arm, the positioning error decreases and so does the learning rate until the system stabilizes at a minimum error and learning rate. This abolishes the need for a predetermined cooling schedule. The automatic cooling procedure results in a "closed-loop" control with no distinction between a learning phase and a production phase. If the positioning error suddenly starts to increase due to an internal failure such as a broken joint, or an environmental change such as a camera moving, the learning rate increases accordingly. Thus, learning is automatically activated and the network adapts to the new condition after which the error decreases again and learning is "shut off." The automatic cooling is therefore a prerequisite for the autonomy and the fault-tolerance of the system.

* This research was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-18605 while the first and second authors were in residence at ICASE, NASA Langley Research Center, Hampton, VA 23665.

1. Introduction

One of the challenges of applying ANNs to the control of autonomous systems is to utilize their ability to learn and to adapt in a way that does not require any outside intervention, such as a teacher, even if drastic failure conditions occur. In order to demonstrate this ability, we implemented an application of an ANN which controls the positioning of a robot end effector by mapping the visual feedback from two cameras to a set of joint angles which are the control signals for the manipulator. This application and the underlying ANN model, called local linear maps [2], is based on the work of Ritter, Martinetz, and Schulten [1–4].

One problem with the approach of Ritter et al. is the need for a predetermined "cooling" schedule which reduces the learning rate over time to stabilize the system after it has learned the required mapping. While the approach works well for the initial learning, it effectively keeps the learning rate at a constant level after the "freezing" of the system, which is insufficient for allowing the network to adapt to sudden and drastic changes in the robot's environment. If the final learning rate is too low, the system takes extremely long to adapt and learns a new mapping poorly. On the other hand, if the final learning rate is too large, the system is able to adapt quickly, but never stabilizes enough to perform precise positioning.

In order to allow the network to react autonomously to different failure events, we propose an automatic cooling procedure that couples the learning rate to the average positioning error which can be obtained from the available camera feedback. This results in a "closed-loop" control with no distinction between a learning phase and a production phase. It enables the network to adapt to changing conditions quickly by raising the learning rate when learning needs to be done and by practically shutting off learning when the robot system is functioning well. In the following three sections, we briefly describe the application, review the approach of Ritter et al., and discuss our modifications. Section 5 illustrates the achievable adaptivity and fault-tolerance of the system by showing how the network automatically recovers from a series of cumulative fault-scenarios.

2. The Robot and ANN Model

Figure 1 shows the "world" of the robot system as seen by an outside observer. The robot arm consists of a single revolute joint at the base and four prismatic joints. Thus, it has five degrees of freedom, two more than is needed to reach all the locations in the workspace. The length of each segment of the arm starting from the base is 0.2, 0.25, 0.2, 0.15, and 0.15.

The robot arm operates in a 3-D workspace determined by the distribution of target points to which the robot end effector should be positioned. The target points are uniformly distributed within a space of 0.7 x 0.4 x 0.2. The distribution of the target locations need not be known *a priori* because the neural network is able to allocate weight vectors to approximate the probability density function of target points within the workspace. The 512x512 pixel images

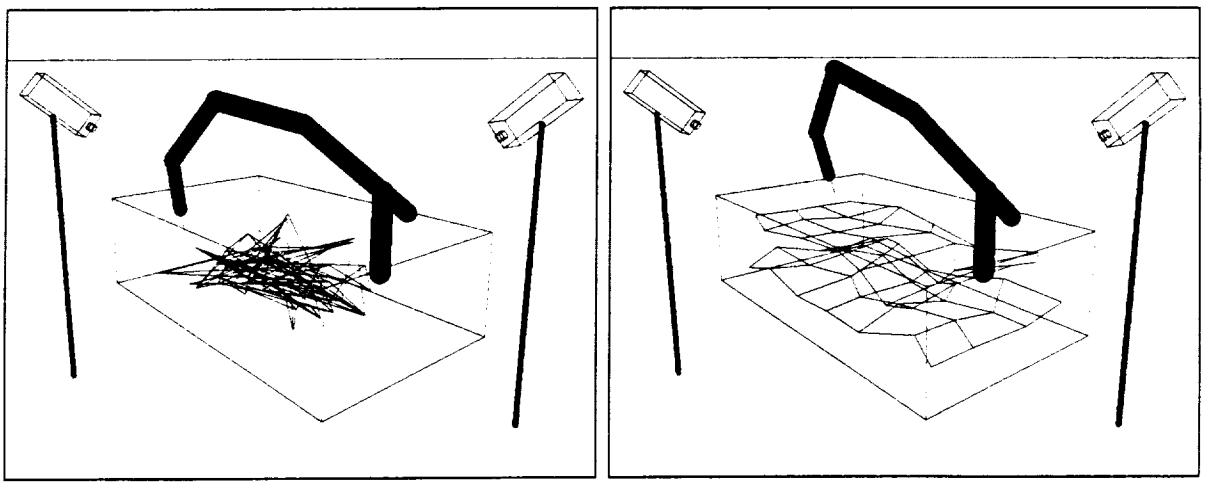

Figure 1. Illustration of the robot and cameras observing the workspace with a representation of the learned mapping before (left) and after (right) training.

provided by the two cameras are processed to yield a 4-D vector $\vec{u}$ which contains the two x-y-coordinates of the target within the workspace.

In contrast to Ritter et al.'s approach [1, 3, 4], we use a local linear map with only 56 neurons which we found sufficient to perform the task. The 56 neurons are connected in a 3–D lattice of 7x4x2 neurons in which each neuron r is responsible for its own nonoverlapping receptive field [1]. Associated with each neuron are the input weights $\vec{w}_r$ and two sets of output weights, $\vec{\theta}_r$ and $\mathbf{A}_r$. Each neuron is connected to the output of the cameras $\vec{u}$ via its input weights $\vec{w}_r$, whereas the output weights $\vec{\theta}_r$ represent the joint angles of the robot arm that should position it at the target location. The additional set of output weights $\mathbf{A}_r$ represent a 5x4 Jacobian matrix which provides a local linear approximation of the mapping $\vec{\theta}(\vec{u})$, valid only in the vicinity of the receptive field of neuron r. This operation, the original learning algorithm, and out modifications are explained in the next section.

3. The Learning Algorithm

Below is a description of the learning algorithm from Ritter, Martinetz, and Schulten [3], except for step 10 which was added to allow "closed-loop" control of the arm. The parameter α is the learning rate and σ determines the shape of a unimodal Gaussian function centered around the "winning" neuron s which determines how much neighboring neurons participate in the adaptation step [3, 4]. Constants α_h and σ_h determine the upper bound of α and σ, while α_l and σ_l determine their lower bound. The moving average of the positioning error is ε. The gain of the function relating the positioning error to the learning rate is λ. The time constants of the low pass filters for the learning rate and the positioning error are β and γ, respectively.

1. The weights of the neural network get initialized to random values and the non-constant parameters get set to their initial values; $\alpha = \alpha' = \alpha_h$, $\sigma = \sigma' = \sigma_h$, and $\varepsilon = 1$[†].

[†] Parameters with a prime are those used with the output weights.

2. A target location is chosen in the workspace. This position is viewed by the cameras giving $\vec{u}$.
3. The neuron whose weights $\vec{w}_r$ are closest to $\vec{u}$ as determined by

$$\|\vec{u} - \vec{w}_s\| = \min_r \|\vec{u} - \vec{w}_r\| \quad \forall r$$

is considered the "winner" and called s.

4. The input weights are immediately updated by

$$\vec{w}_r^{new} = \vec{w}_r^{old} + \alpha h_{rs}\left(\vec{u} - \vec{w}_r^{old}\right)$$

$$h_{rs} = \exp\left(-\frac{\|r - s\|}{2\sigma^2}\right).$$

5. The robot arm changes its angles to $\vec{\theta}_i$ given by

$$\vec{\theta}_i = \vec{\theta}_s + \mathbf{A}_s(\vec{u} - \vec{w}_s^{new})$$

thus moving its end effector. The cameras read the new end effector position $\vec{v}_i$.

6. The robot arm changes its angles again to $\vec{\theta}_f$ given by

$$\vec{\theta}_f = \vec{\theta}_i + \mathbf{A}_s(\vec{u} - \vec{v}_i) \tag{1}$$

and the location of the end effector is read from the cameras giving $\vec{v}_f$.

7. The position can be improved further by replacing $\vec{\theta}_i$ with $\vec{\theta}_f$ and $\vec{v}_i$ with $\vec{v}_f$ in (1) and iterating on step 6, thus doing additional fine movements.
8. Improved estimates of the angle and **A** matrix weights are given by

$$\vec{\theta}^* = \vec{\theta}_s + \mathbf{A}_s(\vec{u} - \vec{v}_i)$$

$$\mathbf{A}^* = \mathbf{A}_s + \frac{\mathbf{A}_s(\vec{u} - \vec{v}_f)(\vec{v}_f - \vec{v}_i)^T}{\|\vec{v}_f - \vec{v}_i\|^2}.$$

9. The output weights of the network are updated with

$$\vec{\theta}_r^{new} = \vec{\theta}_r^{old} + \alpha' h'_{rs}\left(\vec{\theta}^* - \vec{\theta}_r^{old}\right)$$

$$\mathbf{A}_r^{new} = \mathbf{A}_r^{old} + \alpha' h'_{rs}\left(\mathbf{A}^* - \mathbf{A}_r^{old}\right).$$

10. The positioning error in camera space is given by

$$\varepsilon^{new} = \varepsilon^{old} + \gamma\left(\|\vec{u} - \vec{v}_i\| - \varepsilon^{old}\right)$$

and the learning rate and neighborhood size is adjusted based on this error by

$$\alpha^{new} = \alpha^{old} + \beta\left((\alpha_h - \alpha_l)\tanh(\lambda\varepsilon) + \alpha_l - \alpha^{old}\right) \tag{2}$$

$$\sigma^{new} = \sigma^{old} + \beta\left((\sigma_h - \sigma_l)\tanh(\lambda\varepsilon) + \sigma_l - \sigma^{old}\right). \tag{3}$$

11. The learning algorithm now loops back to step 2.

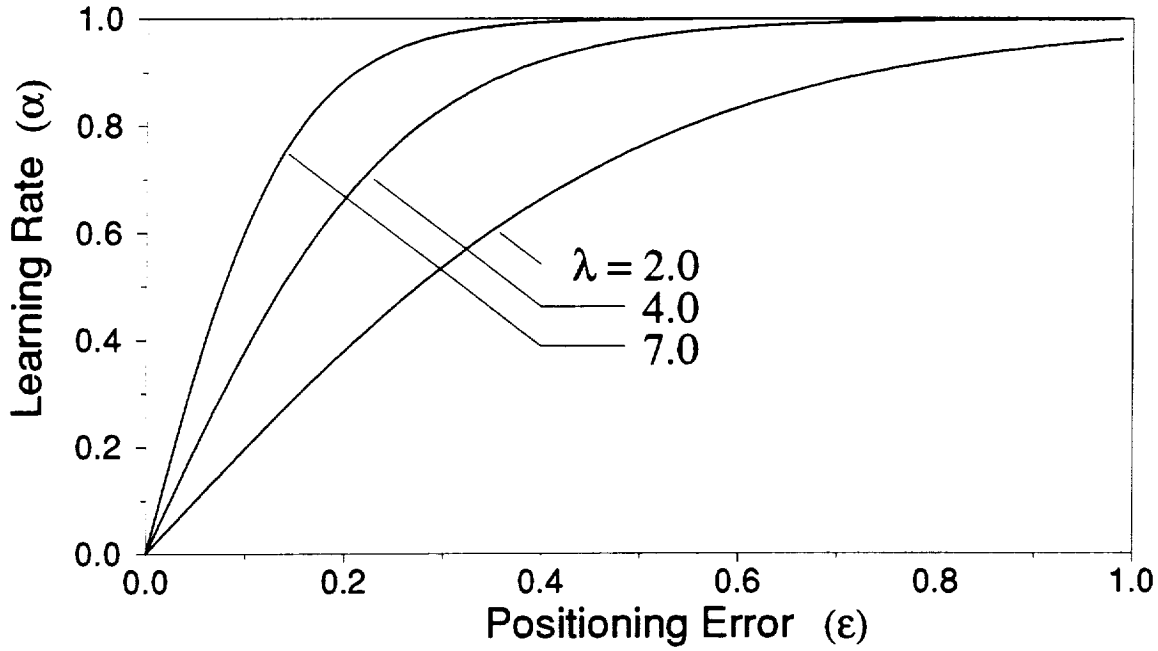

Figure 2. Illustration of the relationship between ε and α for three different values of the gain λ.

4. Description of the Automatic Cooling Procedure

The essence of the automatic cooling procedure is to allow for "closed-loop" control of the learning rate by coupling it to the filtered, average positioning error. In step 5 of the learning algorithm, the first move of the robot arm towards the target is made. With a properly trained network, the difference of the arm position and the target position would be small. The robot does not get any explicit information about the working status of the arm, i.e. if it is functioning properly or not. Hence, if a joint breaks before the move is made, the difference of the arm position and the target position would be large. This will then be reflected in the value of ε, which is the moving average of the positioning error. Because the learning rate is coupled to ε through (2) and (3), the learning rate will increase enabling the network to quickly overcome the fault. Figure 2 shows the relation of α to ε for different gains λ, given that $\beta=\alpha_h=1$ and $\alpha_l=0$. The hyperbolic tangent function in (2) and (3) was chosen because a slight increase in ε when ε is small should result in a large increase in α and σ. It also bounds α to a maximum value of 1.0 which is desirable.

Although the network is not overly sensitive to parameters, instabilities in the network may occur if the gain is too small or too large. If the gain is too small, α will drop too quickly and the network cools too fast, resulting in a poor performance. If the gain λ is too large, the network may never cool. In order for the manipulator to do precise positioning, learning must be local to the individual neuron only, and this can only be done if σ is small (less than 0.7). Thus, the learning rate must be reduced before the positioning error comes down, and the positioning error must come down before the learning rate is reduced. In practice, it is not difficult to find a medium value for λ that results in good performance.

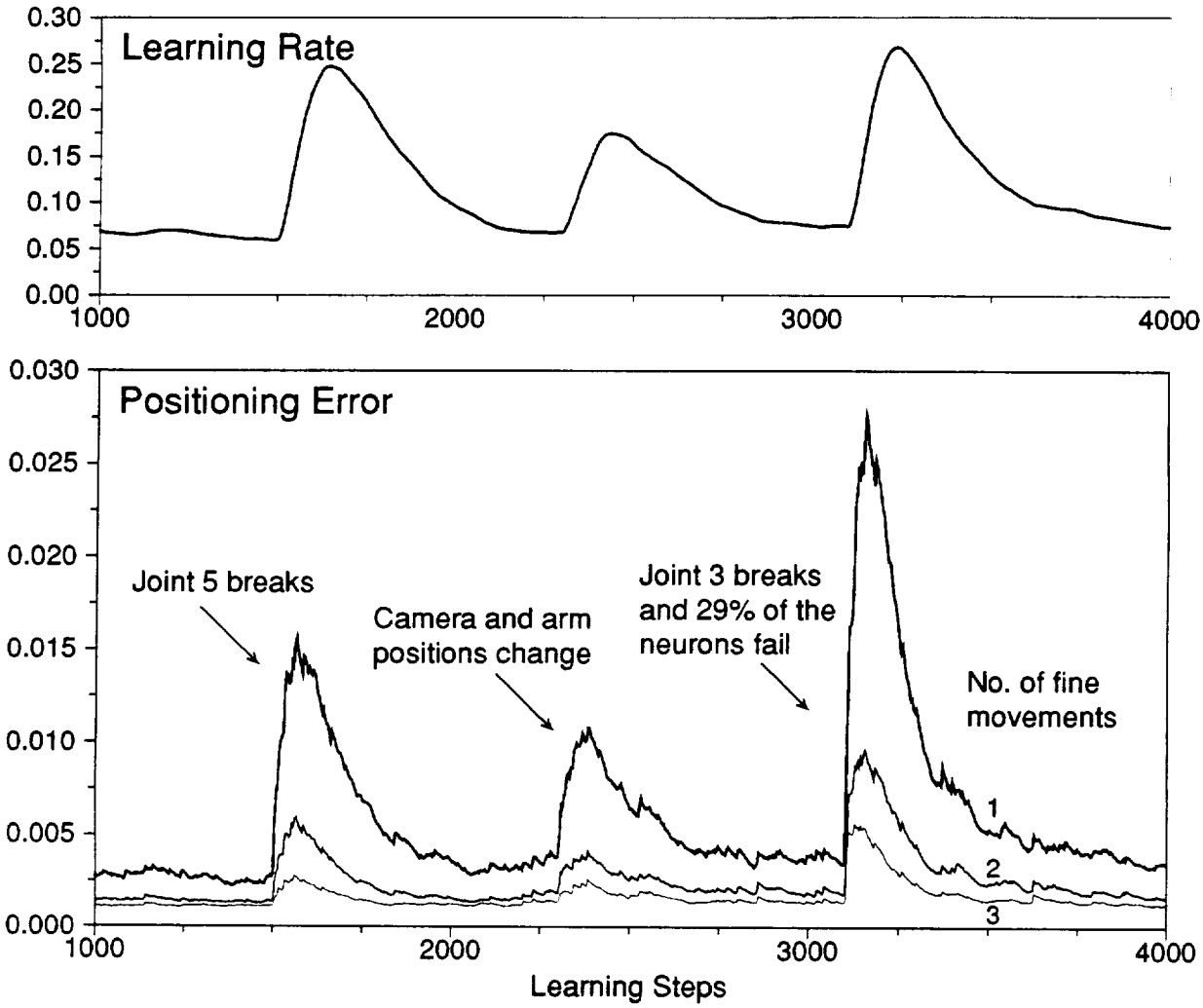

Figure 3. The effect of breaking two joints, moving both cameras, moving the robot, and killing 29% of the neurons can be seen by three separate increases in the positioning error. The effect this has on the learning rate, α, is shown in the upper plot. Notice that the positioning error can be reduced by doing additional fine movements.

5. An Example of the Automatic Cooling Procedure

Figure 3 shows an example of the networks ability to automatically recover from cumulative faults. In this example, initial training consisted of 1,500 learning steps of which the first 1,000 are not shown in Figure 3. During the first 1,000 learning steps the learning rate and positioning error slowly decrease from their initial values set at 1.0, to their values shown at step 1,000 in Figure 3. After 1,500 learning steps joint 5 was broken by "freezing" it at its current position. Immediately, the positioning error increases along with the learning rate, which in turn quickly decreases the error.

After 2,300 learning steps the cameras and the base of the robot are moved with respect to the workspace. It is important to note that when the base position of the robot is moved along

with the cameras, all the weights of the network must change to represent the new geometry of the system.

After 3,100 learning steps, joint 3 is also frozen in its current position. Additionally, a partial failure of the neural network is simulated by "incapacitating" 16 neurons. Of course, this is of practical value only if the ANN is actually implemented in dedicated analog hardware. The loss of the neurons results only in a loss of "resolution" in the overall map. The network is then run until 4,000 learning steps have been completed. Even after all these faults, only a slight performance degradation has occurred, and this can be improved by doing additional fine movements. The parameter values used for the example shown were, $\alpha_h=\alpha'_h=1$, $\alpha_l=0$, $\alpha'_l=0.5$, $\sigma_h=2.5$, $\sigma'_h=1.5$, $\sigma_l=\sigma'_l=0.1$, $\lambda=4$, $\beta=\beta'=0.015$.

6. Conclusion

An automatic cooling procedure for an ANN which controls the positioning of a robot arm has been described. With the use of visual feedback provided by two cameras, the robot system is able to learn to position its end effector anywhere in the workspace. The automatic cooling procedure described is able to automatically activate learning when the positioning error increases allowing the network to adapt quickly to drastic changes in the robot's work environment. This adaptive ability is highly advantageous to conventional robot systems which would require precise recalibration of the robot system components [5].

References

[1] Martinetz, T. M., Ritter, H. J., and Schulten, K. J. Three-dimensional neural net for learning visuomotor coordination of a robot arm. *IEEE Transactions on Neural Networks 1*, 1 (March 1990), 131–136.

[2] Ritter, H. Learning with the self-organizing map. In *Artificial Neural Networks, Vol. 1* (1991), T. Kohonen, K. Maekisara, O. Simula, and J. Kangas, Eds., Elsevier Science Publishing Company, Inc., pp. I–379–384. Proceedings of the 1991 International Conference on Artificial Neural Networks (ICANN-91), Espoo, Finland, 24-28 June, 1991.

[3] Ritter, H., Martinetz, T., and Schulten, K. *Neuronale Netze, 2. Auflage*. Addison-Wesley (Deutschland) GmbH, Bonn, 1990.

[4] Ritter, H. J., Martinetz, T. M., and Schulten, K. J. Topology-conserving maps for learning visuo-motor coordination. *Neural Networks 2*, 3 (1989), 159–168.

[5] Walter, J. A., Martinetz, T. M., and Schulten, K. J. Industrial robot learns visuo-motor coordination by means of "neural-gas" network. In *Artificial Neural Networks, Vol. 1* (1991), T. Kohonen, K. Maekisara, O. Simula, and J. Kangas, Eds., Elsevier Science Publishing Company, Inc., pp. I–357–364. Proceedings of the 1991 International Conference on Artificial Neural Networks (ICANN-91), Espoo, Finland, 24-28 June, 1991.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1992	Technical Memorandum

4. TITLE AND SUBTITLE
Automatic Learning Rate Adjustment for Self-Supervising Autonomous Robot Control

5. FUNDING NUMBERS
307-50-11

6. AUTHOR(S)
Michael K. Arras, Peter W. Protzel, and Daniel L. Palumbo

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23665-5225

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
NASA TM- 107592

11. SUPPLEMENTARY NOTES This research was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-18605 while the first and second authors were in residence at ICASE, NASA Langley Research Center, Hampton, Virginia 23665

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Unclassified-Unlimited

Subject Category 63

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

Described is an application in which an Artifical Neural Network (ANN) controls the positioning of a robot arm with five degrees-of-freedom by using visual feedback provided by two cameras. This application and the specfic ANN model, local liner maps, are based on the work of Ritter, Martinetz, and Schulten. We extended their approach by generating a filtered, average positioning error from the continuous camera feedback and by coupling the learning rate to this error. When the network learns to position the arm, the positioning error decreases and so does the learning rate until the system stabilitzes at a minimum error and learning rate. This abolishes the need for a predetermined cooling schedule. The automatic cooling procedure results in a "closed-loop" control with no distinction between a learning phase and a production phase. If the positioning error suddenly starts to increase due to an internal failure such as a broken joint, or an environmental change such as a camera moving, the learning rate increases accordingly. Thus, learning is automatically activated and the network adapts to the new condition after which the error decreases again and learning is "shut off." The automatic cooling is therefore a prerequisite for the autonomy and the fault-tolerance of the system.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Artificial Neural Network, Closed-loop	7
	16. PRICE CODE A02

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified		